JPL
IN-12-CR
260646
45P.

JPL Publication 89-25

# A Planning and Scheduling Lexicon

Jennifer W. Cruz
William C. Eggemeyer

September 15, 1989

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 89-25

# A Planning and Scheduling Lexicon

Jennifer W. Cruz
William C. Eggemeyer

September 15, 1989

NASA

National Aeronautics and
Space Administration

**Jet Propulsion Laboratory**
California Institute of Technology
Pasadena, California

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

## ABSTRACT

This publication focuses on mission planning and scheduling for spacecraft. Planning and scheduling work is known as *sequencing*. Sequencing is a multistage process of merging requests from both the science and engineering arenas to accomplish the objectives defined in the requests. The multistage process begins with the creation of science and engineering goals, continues through their integration into the sequence, and eventually concludes with command execution on board the spacecraft.

The objective of this publication is to introduce some formalism into the field of spacecraft sequencing-system technology. This formalism will make it possible for researchers and potential customers to communicate about system requirements and capabilities in a common language. This publication is not intended to be the definitive work describing all capabilities and requirements; it is meant to lay a good enough foundation to allow follow-on development.

## ACKNOWLEDGMENTS

The authors would like to thank the members of the Sequence Automation Research Group for their assistance and support: Sven Grenander, Mark Rokey, Steve Peters, Win Lombard, Bill Dias, John Sisino, Carol Collins, Elaine Zamani, Mike Hollander, David Mittman, Russ Brill, Tom Starbird, and Julia George.

TABLE OF CONTENTS

# SECTION 1

# INTRODUCTION

## 1.1 PURPOSE

Planning and scheduling techniques are used in many different arenas. It is the objective of this publication to introduce a lexicon that will make it possible for workers to have a common terminology for discussing planning and scheduling issues.

## 1.2 SCOPE

This publication focuses on mission planning and scheduling for spacecraft. Planning and scheduling work is known as *sequencing*. We do not cover areas such as robotic spatial planning and job shop scheduling.

## 1.3 ORGANIZATION

This publication first discusses the stages involved in spacecraft sequencing (Section 2), and continues with discussions of sequencing elements (Section 3) and classification of resources (Section 4). Following are the dependencies that are involved in spacecraft sequencing (Section 5), and a section on the constraints and considerations that should be observed when sequencing is to take place (Section 6). The subsequent section (Section 7) is about the measurement of sequences. A glossary of terms (Section 8) and a list of references (Section 9) conclude this publication.

# SECTION 2

# SEQUENCING

*Sequencing* is a planning and scheduling process. Usually incremental and iterative, it resolves requested activities that are used for accomplishing specific objectives into a temporal plan. In this process, the objectives that cannot be met are either deleted or revised. If enough of the objectives cannot be met, then either the activities supporting the objectives or the objectives themselves must be changed [1]. The following sections describe the terminology associated with the sequencing process.

## 2.1 SEQUENCE

A *sequence* is the result of the sequencing process. It is a product representing the final temporal plan of actions that, once performed, will accomplish the specific objectives. These actions are usually in the form of specific commands. The sequence product may exist as a time-ordered listing, an integrated timeline, and/or a program set of mnemonics [2].

## 2.2 STAGES OF SEQUENCING

There are eight stages in the sequence-development process. The following sections describe these stages in detail. Note that while a sequence is being generated, two or more of these stages may be executed simultaneously. Some of these stages may also loop back on themselves [3].

### 2.2.1 Sequence Temporal Definition

*Sequence temporal definition* is the process of defining the temporal location and the duration of a *load* (sequence). (Note: A spacecraft mission may have many loads throughout the lifetime of the mission.) This definition process is performed before the actual sequence-generation process and is usually performed only once. The sequence temporal definition is usually dictated by the following two considerations:

(1) *Intervals.* The Deep Space Network (DSN) sequences are generated for one-week time intervals. The one-week intervals were selected because the users/sequencers work in one-week intervals [4,5].

(2) *Constraints and Activities.* Sequence start and stop times are based upon constraints and the number of activities to perform. The Voyager spacecraft, for example, has pre-determined start and stop times for its sequences. These times are based upon both the limitation in the number of commands the spacecraft can store and on the amount of activity the spacecraft will perform during its sequence execution. During cruise, there are fewer observations than when the spacecraft is close to a planet, so the amount of activity occurring on board the spacecraft is less; therefore, the load may be four weeks long. When the spacecraft is close to a planet, a number of unique observations are made in a short amount of time. In the latter case, this increase in the temporal density of activity causes loads to be one to three days long [6].

### 2.2.2 Information Gathering

*Information gathering* is the beginning of the cycling, or reiteration, process for sequence development. The information-gathering process occurs when the requesters (people who are requesting that an experiment/activity be performed on board the spacecraft) gather information about the *resources* that are required to perform desired experiments. Initially, information consists of facts about available resources, uplink/downlink channels, and configurations of the spacecraft. But, as the sequence-generation process cycles, information gathering will include not only spacecraft information but also information from the analysis of the data from previously executed experiments [7]. *Opportunity analysis* (which tells where in time the experiment is to take place) is also included in this stage. The data accumulated from information gathering goes into the next stage, *request generation.*

### 2.2.3 Request Generation

This process generates *requests* that fulfill the objectives of the experiment. The request consists of the objectives of the experiment, a description of the experiment, and information about the resources to be used and the frequency and temporal location of the experiment. The information acquired from the information-gathering stage is used in generating the request. Both the information-gathering stage and the stage that follows it, *request and world translation/representation*, may be executed concurrently with the request-generation stage [3].

### 2.2.4 Request and World Translation/Representation

After a sequencing tool is selected to assist in sequence generation, the request and world translation/representation stage translates the request from the request-generation stage into a format that a sequencing tool understands. This is the process of translation from the "human-readable" request down to a machine-input format (i.e., a change in link request; see both "link" entries in the Glossary). The request is broken down into a more detailed level of actions that must be performed to accomplish the objective of the experiment. These detailed actions take the forms indicated in the *sequencing elements* (see Sec. 3, "Sequencing Elements"). On many occasions, this stage is merged with the request-generation stage.

### 2.2.5 Request Integration

The *request integration* stage is the process of scheduling all of the requests or the detailed actions of a request. Start and stop times are assigned to the requests, along with the requests' resource allotments (see Sec. 4, "Resources"). All of these requests that use spacecraft resources must be scheduled within the maximum resource-allotment period, and all sequencing constraints must be enforced. During this phase, a request may undergo changes that will help eliminate conflicts (resource oversubscription and constraint violations are examples of conflicts) in the sequence. The product from this stage is a sequence. This phase is usually the most expensive and time-consuming portion of the overall sequencing process [3,8].

### 2.2.6 Sequence Translation

*Sequence translation* is usually a batch-oriented process of direct expansion from the derived sequencing mnemonics into the explicit machine code commands to be executed by the spacecraft. This stage exists only for sequences that pertain to the operation of remote machines. Included in some of these sequences are expanded *ground commands*, which are detailed instructions to ground support personnel. This stage may also be merged with the *sequence validation/simulation* stage [3].

### 2.2.7 Sequence Validation/Simulation

The sequence validation/simulation stage is a detailed constraint check of the preliminary sequence, which is generated during the request integration stage. All or selected portions of the sequence may be simulated to ensure spacecraft and instrument safety along with sequence integrity. (See Sec. 7, "Sequence-Quality Determination.")

### 2.2.8 Sequence Execution

This process is the actual execution of the sequence. The sequence is executed by man (i.e., a mission specialist or pilot), machines (i.e., instruments), or a combination of the two. Information gained from sequence execution will influence and impact future sequences [6].

The remaining portions of this publication will focus on the terminology relating to three stages: request generation, request and world translation/representation, and request integration.

# SECTION 3

# SEQUENCING ELEMENTS

The term *sequencing element* refers to actions, requests, and activities that exist at different *abstraction levels* in the planning process. Abstraction levels refer to the levels of detail at which the sequence elements exist. Below are descriptions of various types of sequencing elements used in the planning process. See Figure 1, "Sequencing Elements," for a hierarchical diagram.

## 3.1 GOALS

The engineers' objective is to maintain the spacecraft's health and safety and keep the spacecraft at its optimum performance level. The scientists have experiments they would like to perform on board the spacecraft to meet certain science objectives. Both the engineering and science objectives are called *goals.* A goal is an objective to accomplish or perform on board the spacecraft.

## 3.2 INPUT REQUESTS

An *input request* is the result of the request-generation stage. The input request, or request, consists of the objective of the experiment, a description of the experiment, the resources to be used, the desired temporal location, the duration of an observation, the frequency of performing the experiment, and other information of this type. Input requests fall into two categories: *specific requests* and *repetitive requests* [5,9].

### 3.2.1 Specific Requests

Specific requests pertain to requests of a particular type and a particular time specification. Specific requests are for performing unique objectives in a sequence.

*Example 1. A Picture Request*

| | |
|---|---|
| Name: | Red Spot picture |
| Objective: | Gather information on Jupiter Red Spot |
| Description: | Take one picture of the Jupiter Red Spot |
| When: | One day before closest approach |
| Frequency: | Only once |
| Resources: | |

| | |
|---|---|
| Target: | Jupiter Red Spot |
| Instrument: | Narrow-angle camera |
| Data Rate: | 115 kbits/sec |
| Scan Platform Activity: | 15 degrees azimuth/113 degrees elevation |
| Duration of Observation: | 3 min |

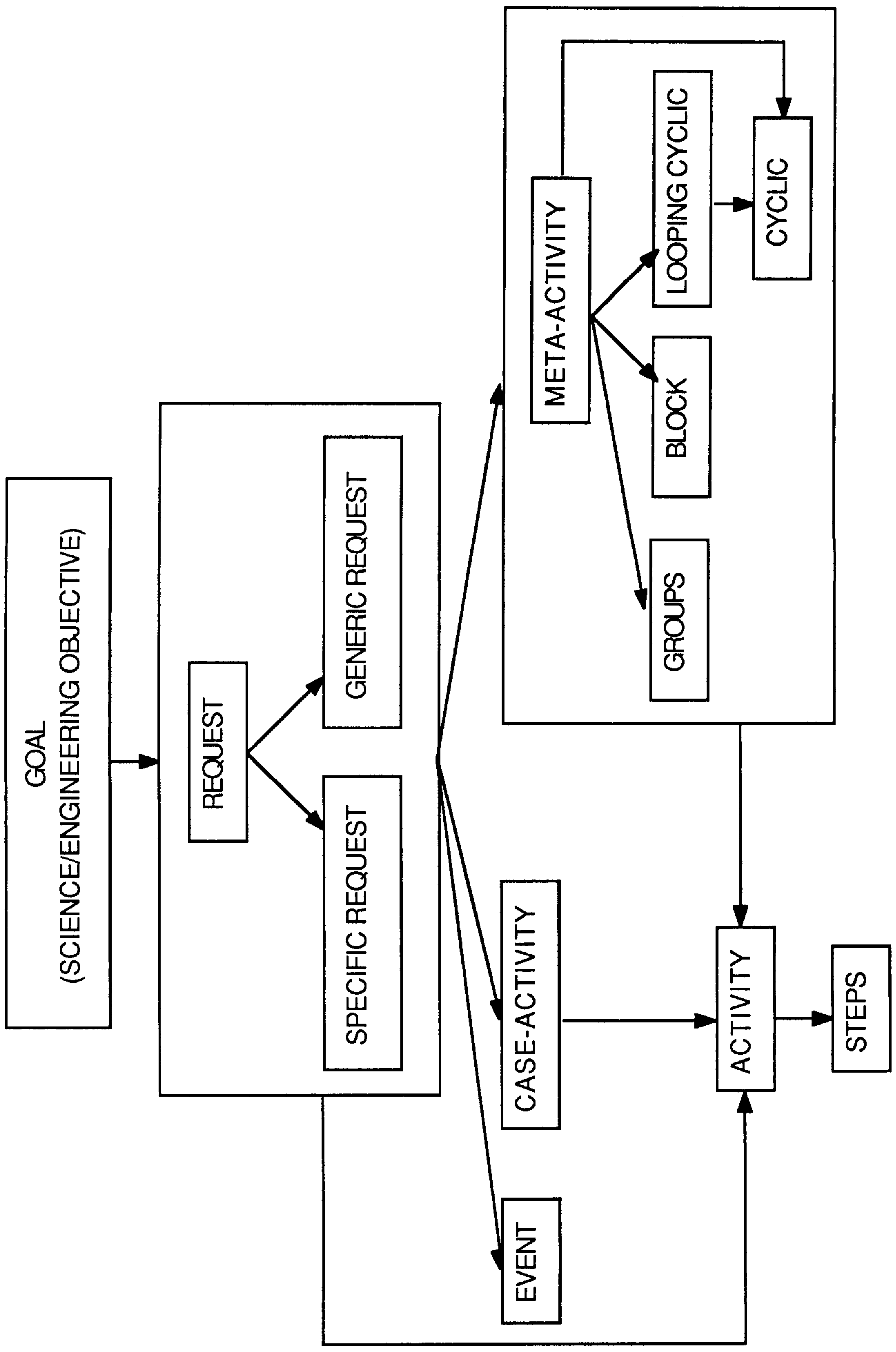

Figure 1. Sequencing Elements

### 3.2.2 Repetitive Requests

Repetitive requests are multiple executions of the same experiment in the sequence. Usually there are temporal relationships between the multiple occurrences of the request [10].

*Example 2. Dummy Sequence Request for CRAF*

| | |
|---|---|
| Name: | WHIGH |
| Objective: | To characterize plasma environment |
| Description: | To obtain high-rate plasma wave data |
| Target: | None |
| Instrument: | Plasma wave analyzer |
| Data Rate: | 115.2 kbits/sec |
| Scan Platform Activity: | None |
| Duration of Observation: | 10 min |
| Frequency: | Once a track (per DSN coverage) |

## 3.3 REQUEST STRUCTURES

After a request is generated from a goal, the request is broken down to a more detailed level. The lowest structures at the detailed level are *independent events* and *steps.*

### 3.3.1 Independent Events

There are two kinds of independent events. The first is a stand-alone structure that has no other events dependent upon it. It contains information such as start/stop times and resource usage. For example, on the shuttle there are crew members. A crew member sleeping is an independent event, because when the crew member is sleeping, there are no other actions dependent upon him.

An independent event can also be a description of the world knowledge or resource allocation. For example, a sequence for an orbiting spacecraft might have a sequencing element that represents the orbit's periapsis. This element is an *informational event* and is not generated from the goals or requests. Such events, which are based upon knowledge of the spacecraft's environment or state, are placed into the sequence products to provide more information for reviewers of the sequence.

### 3.3.2 Steps

A step is very similar to an independent event, but steps may have other steps dependent upon them. Steps represent actions that are to occur in the sequence. Steps may have both temporal resource dependencies and temporal step dependencies. For example, suppose the goal is to take a picture with an automatic camera. To achieve this goal, we have to perform two actions. The first step is to position the camera, and the second is to shutter it. The temporal step dependency is that the camera positioning must take place before shuttering. The resource dependency is that the camera must be available for both positioning and taking the picture.

## 3.4 ACTIVITIES

An *activity* is a collection of one or more actions to perform in order to accomplish the goal or objective. These actions take the form of steps with temporal resource dependencies (see Sec. 4, "Resources") and temporal step dependencies (see Sec. 5, "Dependencies") [2]. Let's look at the example in the "Steps" section (above). The two steps described above, (1) positioning the camera and (2) shooting the picture, would be encompassed by an activity structure.

Activities are the basis for the more complex sequencing elements. Built on activities are the *case-activity, meta-activity, block,* and *cyclic.* Below is a description of each of these structures [11].

### 3.4.1 Case-Activity

A case-activity is an activity that has multiple step configurations. All of these step configurations are predefined. The actual configuration of steps that is used may be determined when the sequencing actions are being performed, or the user may select which configuration to use. The step linkages in case-activities are based upon IF-THEN-ELSE constructs. To date, case-activities have not been implemented on any project, but the Mars Rover, because of its dynamic environment, would be a valid sequencing arena in which to use a case-activity. Let's examine a Rover example. The goal is to pick up a sample and place the sample into a canister. One step configuration might be

```
TOP
        Pick Up Sample
IF didn't acquire sample, ask for help
        ELSE
        IF Sample is big
                THEN Place sample in big canister
                ELSE Place sample in small canister
```

In a case-activity, all possible configurations are predefined, and selection of which configuration to use is based upon the resource availability at the time when the actions are to occur. In the above example, the resource (canister) availability is focused upon whether or not the sample was picked up.

### 3.4.2 Meta-Activity

A meta-activity consists of multiple occurrences of the same activity in the sequence. Meta-activities are based on a structural implementation methodology. Meta-activities may contain temporal dependencies. When the same activity is called many times throughout the sequence, it is practical to create the activity structure once (in the manner of a subroutine) and call it whenever the activity is to be performed. If this is done, then when the activity is to occur, the activity function is invoked by just a function call. Using meta-activities results in memory savings: Less memory is required for one activity structure plus many calls to the structure than for duplicate instances of the same structure in the sequence. There are four types of meta-activities: *cyclics, looping cyclics, blocks*, and *groups.* These four types are described below.

3.4.2.1 Cyclics. A cyclic is a meta-activity consisting of an activity that can be invoked multiple times within the sequence. The step configuration within the activity does not change. Cyclics may be redefined from sequence to sequence. In the following example, assume that our objective is to record data and that we will record data the same way each time [6].

*Example 3. Cyclics*
For the Voyager spacecraft, a data-mode activity requires four CCS words. To create a data-mode cyclic requires the initial four CCS words plus one additional word, for a total of five CCS words for the cyclic. Each call to the cyclic is two CCS words. If a non-cyclic data-mode activity is executed three times, a total of 12 CCS words is required. But if the activity is a cyclic, then the cost is the initial five CCS words for creating the cyclic, plus three calls at two CCS words per call, for a total of 11 CCS words. So in this example, the data-mode cyclic will be cost-effective if the cyclic is called a minimum of three times.

3.4.2.2 Looping Cyclics. A looping cyclic is a special form of a meta-activity. Looping cyclics define a temporal relationship among the multiple executions of a specific type of activity. Usually, the looping cyclic specifies the number of times to perform an activity and a specific amount of time between each execution. Looping cyclics may be redefined from sequence to sequence. For example, see activity ABC (Figure 2, below), which performs five times, with each execution separated by a Δt of one day.

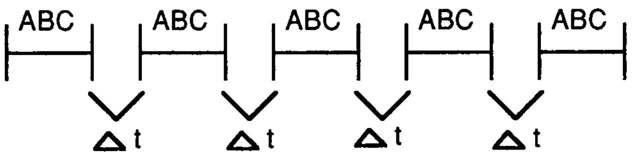

Figure 2. A Looping Cyclic

3.4.2.3 Blocks. A block is a specific type of meta-activity; it can be invoked more than once in multiple sequences. Usually, a block is defined at the beginning of the mission and the sequence of steps in the block does not change; however, the duration of the steps may change from load to load. Blocks are given parameters that define the duration of their steps. These parameters are based upon the state of the sequence. An example is the Voyager Trajectory Command Maneuvering Block (TCM). The duration of this type of block is dependent upon the trajectory. The larger the trajectory change, the longer the duration of the block. The TCM is used throughout the mission, and its steps change only in duration [6].

3.4.2.4 Groups. Groups are collections of steps and/or activities. Groups have a twofold purpose. They catalog the sequencing elements for informational purposes. They also apply additional contextual control (see Glossary) over the sequencing actions that occur in the group. (Note that Project Galileo uses the term "groups" in connection with CDS memory management. The Galileo term is not analogous to the definition given here.)

# SECTION 4

# RESOURCES

A *resource* is a source of supply, support, or availability that is required for performance of a given task. Resources cover a wide variety of areas. Examples of different resources are power consumption, instrument usage, telemetry mode, crew member usage, food consumption, trajectory and pointing orientation, camera shutter settings, instrument on/off status, environmental changes, and target-in-view.

Resource usage, or *subscription*, is one of the constraints that must be considered in sequencing. When a task is scheduled, resources must be allocated to perform the given task. But more times than not, many tasks are competing for the same resources in the same time interval. In spacecraft scheduling, it's typical to have more tasks requested than resources available to perform them [12,13].

The resources in the spacecraft arena fall into different classes: *synchronicity resources, non-depletable resources, depletable resources, state resources,* and *special resources* (sometimes called *hybrid resources*) such as a data tape recorder. This section covers the different types of resources that are scheduled in the spacecraft arena, along with their usage and conflict definitions.

## 4.1 SYNCHRONICITY RESOURCES

Availability, or synchronicity, resources represent the existence or non-existence of an item or action [3]. Sample resource types include target-visibility status, spacecraft-maneuvering status, environmental effects, and environmental constraints.

The current state of an availability resource is determined by two means, the first of which is spacecraft and/or instrument orientation.

*Example 4. A Synchronous Resource*
An experiment requests to take a picture of Target X. (The resource is Target X.) When Target X is in view of the instrument, the current state of the Target X resource is "in view." At other times, the state of Target X is "not in view." The availability of a target is determined by both the orientation of the spacecraft and by the instrument-pointing direction.

*Example 5. A Non-Synchronous Resource*
A second example is that of performing a micro-gravity experiment. This kind of experiment must be performed when the spacecraft is not maneuvering. The resource is spacecraft maneuvering. If the spacecraft is not maneuvering, the current state of the time frame is "not maneuvering" and the experiment may be performed (if this decision is based solely on the spacecraft maneuvering resource). When the spacecraft is maneuvering, the current state is "maneuvering" and the experiment cannot be performed. This is another example of the spacecraft condition determining the current state of a resource.

The current state of an availability resource may also be based upon the side effects of an experiment, as the next example illustrates.

*Example 6. A Combination of Synchronous Resources*
An imaging experiment takes three mosaics of Target Z. Imaging requires that during camera shuttering, the platform cannot jitter nor can the spacecraft thrusters fire. The platform jitter would cause the pictures to be out of focus, and the thruster firing would result in outgassing, which would obscure the camera field-of-view. The two resources in this example are jitter and outgassing. The current state of these resources is determined by the experiments being performed in a given time frame. Let's say that Experiment A is in the same time frame as the

imaging experiment. Experiment A uses the tape recorder, which indirectly causes outgassing. This outgassing is the result of thruster firings compensating for the tape recorder's torquing of the spacecraft. So now the current state of the jitter resource is "jitter" and the current state of the outgassing resource is "outgassing." This means that the imaging experiment cannot take place in the same time frame as Experiment A. This is an example of experiments setting the current states of availability resources. The example also demonstrates a temporal non-synchronous type of dependency between the two experiments.

To sum up: The availability resource state is determined by spacecraft and/or instrument orientation and by the side effects of experiments. Other experiments' usage of availability resources may be affected by these current states. Conflicts for availability resources will occur in the sequence if the current states of the resources do not coincide with the states requested by the experiments for a given time frame.

## 4.2 NON-DEPLETABLE RESOURCES

A non-depletable resource is one that can be utilized to its maximum allocation during a time frame, independent of the use at other times. Non-depletable resource usage is cumulative over all users of that resource during a time frame. After a task concludes, the non-depletable resource that the task utilized is again available for use by other tasks [3,11,14]. (See the following examples.)

Sample resource types include crew member usage, instrument usage, power usage, and thermal usage.

Synonymous terms for "non-depletable" are "pooled," "non-consumable," "reusable," and "pointwise."

### 4.2.1 Usage

*Non-depletable resource usage* is the amount of a non-depletable resource that a sequencing element will consume or utilize. If more than one experiment is active at specified times, then the total resource usage for those times is the summation of all the experiments' use of the resource.

*Example 7. Usage*
Assume that Instrument A is in use and that it requires 20 watts of power. At the same time, a second instrument is operating and requires 15 watts of power. The total resource usage for that time is 35 watts of power. Notice that the sequencing elements are users of the resource, but are not "aware" of the total resource usage.

### 4.2.2 Maximum Allocation

*Maximum allocation* is the available amount of a resource. Maximum allocation may be a constant, predefined distribution with temporal dependencies; may change temporally, depending on a mathematical formula like a step or sinusoidal function; or may rely on other actions occurring in the sequence [11].

*Example 8. Predefined Allocation*
A sample table for the maximum allocation of ac power is

| TIME | ALLOCATION |
|---|---|
| 1–2 | 20 watts |
| 3–5 | 25 watts |
| 5–7 | 15 watts |
| 7–10 | 20 watts |

This chart shows that from time 1 to time 2, the power allotment is 20 watts; from time 3 to time 5, the power allotment is 25 watts; from time 5 to time 7, the power allotment is 15 watts; and from time 7 until the end of the sequence, the power allotment is 20 watts. This is an example of a constant, predefined distribution.

*Example 9. Temporal Allocation*
A sample table for maximum allocation that changes as a function of time is

$$\text{Power} = 0.5\,(\text{time} \cdot 3) = 1.5\ \text{time}$$

| TIME | ALLOCATION |
|---|---|
| 1 | 1.5 watts |
| 3 | 4.5 watts |
| 5 | 7.5 watts |
| 10 | 15.0 watts |

This chart illustrates allocation as a function of time.

*Example 10. Derived Allocation*
A solar cell experiment performed on Earth is dependent upon cloud coverage. The more clouds present, the less power generated. The fewer clouds present, the more power generated. In this example, we see that maximum allocation (generation of greatest possible power) is a function of other activities (cloud coverage), in the sequence.

### 4.2.3 Oversubscription

*Non-depletable resource oversubscription* is the amount by which the resource requested exceeds the maximum allotment for a given time frame. If the resource is oversubscribed in a given time frame, that time frame is flagged as a conflict area.

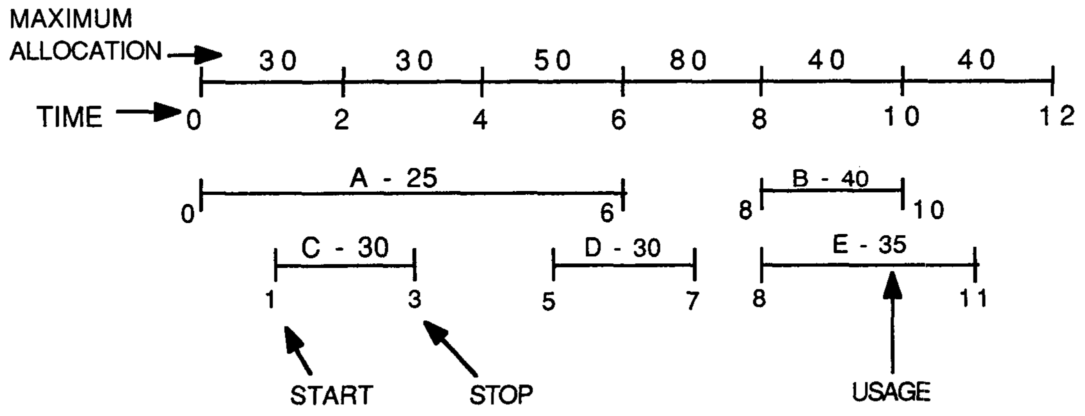

Figure 3. A Non-Depletable Resource: Power

*Example 11. A Non-Depletable Resource: Power*
Figure 3 shows five sample experiments requesting power. The top portion of the figure is a timeline, with the maximum power allocations corresponding to the time frames. Below the timeline are five tasks, with their start and stop times and their requested power usage. Power is a cumulative resource; the breakdown is given by Table 1 ("Power-Profile Table"). An oversubscription of greater than zero represents conflict in the schedule for the given time span.

Table 1. Power-Profile Table

| Time Span | Max. Allocation | Total Usage | Oversubscription |
|---|---|---|---|
| 0-1 | 30 | 25 | 0 |
| 1-2 | 30 | 55 | 25 |
| 2-3 | 30 | 55 | 25 |
| 3-4 | 30 | 25 | 0 |
| 4-5 | 50 | 25 | 0 |
| 5-6 | 50 | 55 | 5 |
| 6-7 | 80 | 30 | 0 |
| 7-8 | 80 | 0 | 0 |
| 8-10 | 40 | 75 | 35 |
| 10-11 | 40 | 35 | 0 |

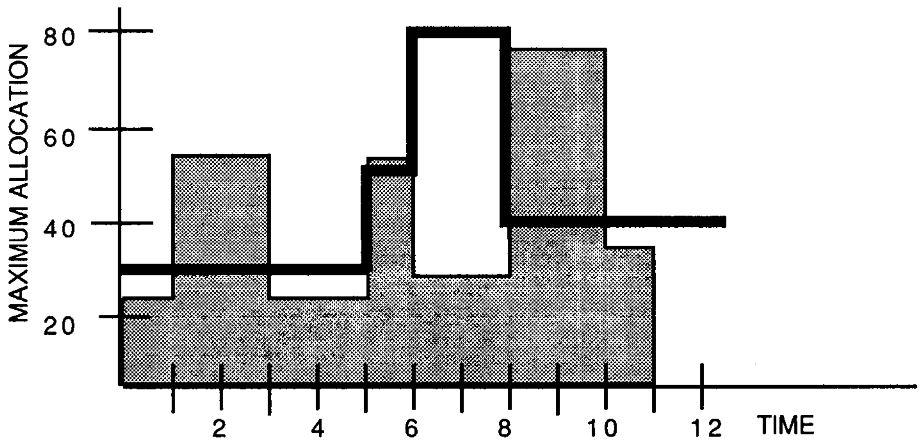

Figure 4. A Non-Depletable Resource Profile

Figure 4 ("A Non-Depletable Resource Profile") is a graphical representation of Table 1 ("Power-Profile Table"). The solid black line represents the maximum resource allocation. The gray areas represent the total resource usage. The gray areas above the solid black line represent resource oversubscription.

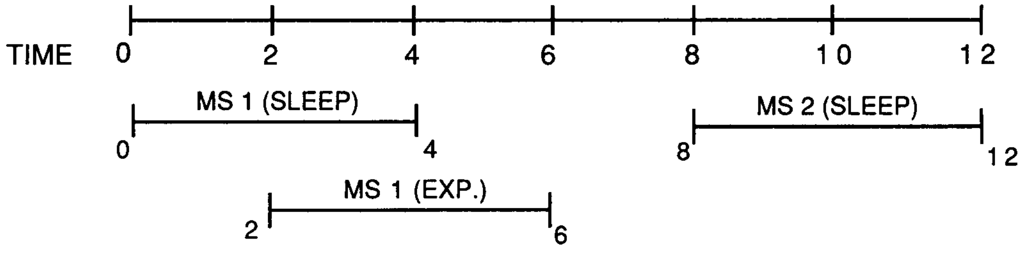

Figure 5. A Simple Non-Depletable Resource: Crew

*Example 12. A Simple Non-Depletable Resource: Crew*
Crew members are another example of a simple non-depletable resource. In the crew usage example (Figure 5), two crew members are requested to perform three tasks. The three tasks are Mission Specialist 1 sleeps between times 0 and 4, Mission Specialist 1 performs an experiment between times 2 and 6, and Mission Specialist 2 sleeps between times 8 and 12. Obviously, Mission Specialist 1 will have a conflict during times 2 to 4 because he can either sleep or perform

the experiment, but not both. In some instances, a crew member can perform more than one task at a time, depending on the task.

## 4.3 DEPLETABLE RESOURCES

Depletable resources are consumed as they are used. The depletion rate is dependent on both time and the rate of consumption. Some depletable resources are resettable; others are replenishable. Depletable resources are distinctly different from non-depletable resources, since depletable resources maintain a running total of the resource usage. At the commencement of a task, the total resource allotment does not change. When the usage exceeds the maximum allocation, a conflict occurs. At this time, either the resource must be replenished, or a scheduling action must occur to alleviate the conflict [14].

Sample resource types include energy, fuel, food, and CCS words (not including those affecting memory management). Synonyms for "depletable" are "consumable" and "cumulative."

*Example 13. A Depletable Resource: Food Supply*
Food is a depletable resource. For a one-week Space Station mission, the food supply is approximately 7 pounds of food per day for each crew member. The mission has six crew members, so a total of 294 pounds of food is available for a one-week mission. At the start of the mission, the crew members eat 6 pounds of food per day because they're adapting to the types of food and to their environment. But toward the middle of the mission, the members start to consume more than their 7 pounds-a-day allotment. If the total food consumption exceeds 294 pounds before the seven days are concluded, then the crew members will go hungry until the food supply is replenished. Below is a graphical depiction of this example.

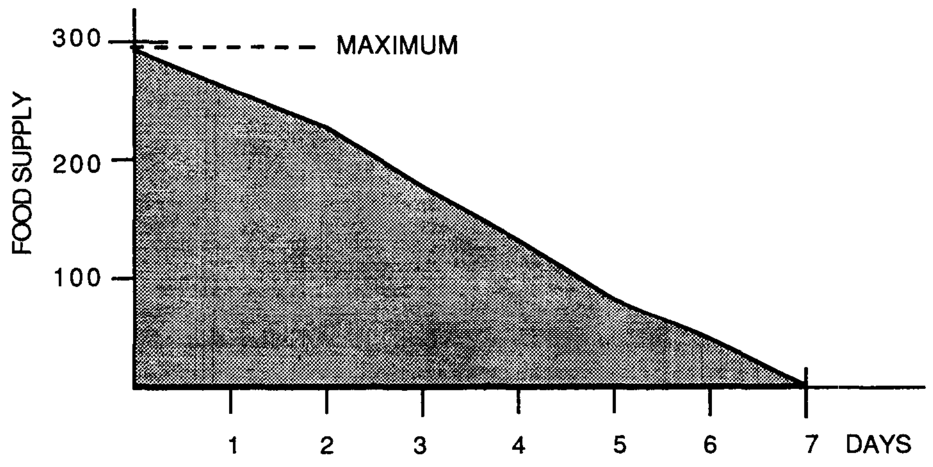

Figure 6. A Depletable Resource: Food Supply

Figure 6 ("A Depletable Resource: Food Supply") shows the food supply depleting as the crew members consume food. The food supply is directly dependent upon the rate of consumption. The less food the crew members consume, the slower the food supply depletes. Note that the food supply cannot be replenished until a supply shuttle rendezvous with the Space Station.

## 4.4 STATE RESOURCES

A *state* is a mode or a condition of being. Some tasks set the actual mode of an instrument, while other tasks just check to see if they can use the instrument in its current mode. Other tasks prohibit certain states from occurring within selected time frames.

Sample resource types include data mode settings, instrument settings, and trajectory settings.

Tasks that are involved with state resources have been classified into three types of scheduling objects: *changers*, which set the actual state/mode; *users*, which use whatever state is available; and *prohibitors,* which prohibit certain states from occurring. Below is a description of the state scheduling objects.

### 4.4.1 Changer Scheduling Objects

Changer objects set or change the actual state of a resource. Only changer objects are allowed to change the actual state. The resource state that is set by the changer object will remain constant until another changer object is scheduled to change the resource state.

*Example 14. A State Changer*
Task A turns on Instrument A with Mode ZZ at time 5 and uses Instrument A until time 6. Task B wants to use Instrument A at time 7, but in Mode YY. So Task B checks whether Instrument A is ON or OFF and also checks the mode setting. Seeing that Instrument A is still ON, but in the incorrect mode setting, Task B changes the mode setting to Mode YY and proceeds with its experiment.

Example 14 shows that the changer object, Task A, turned ON Instrument A with Mode ZZ. But the Task A time frame was from 5 until 6. At the conclusion of Task A, Instrument A still remained ON with Mode ZZ until another task was scheduled that would change the mode setting of Instrument A. To reiterate, once a changer scheduling object sets the actual state, the actual state will remain constant, even after the conclusion of the task, until another changer scheduling object that will change the actual state is scheduled. Note that changer scheduling objects may also be considered users of the actual state after the actual state has been set.

### 4.4.2 User Scheduling Objects

User objects "use" the actual state that the changer object has defined. User objects do not change the actual state.

Refer to Example 14 above and assume that Task B still wants to use Instrument A with Mode YY, but that Task B is only a user object. This means that Task B checks whether Instrument A is ON or OFF. Instrument A is ON, so now Task B checks whether Instrument A is in Mode YY. Task B confirms that Instrument A is not in Mode YY but is currently in Mode ZZ. If possible modes for Task B to use include Mode ZZ and not the desired mode, Mode YY, then an additional changer object is not needed.

### 4.4.3 Prohibitor Scheduling Objects

Prohibitor objects define non-valid states. When a prohibitor object is scheduled, the states on the list contained by the prohibitor object are unusable during the prohibitor object's time frame. All of the states in the prohibitor's state list are unschedulable states.

### 4.4.4 Conflict

There are two types of conflict for state resources. The first type of conflict occurs when a user scheduling object's potential states do not match the actual state defined by a changer scheduling object. The other type of conflict occurs when a prohibitor scheduling object prohibits a particular state in a given time frame, and either a changer or a user scheduling object is scheduled in that time frame.

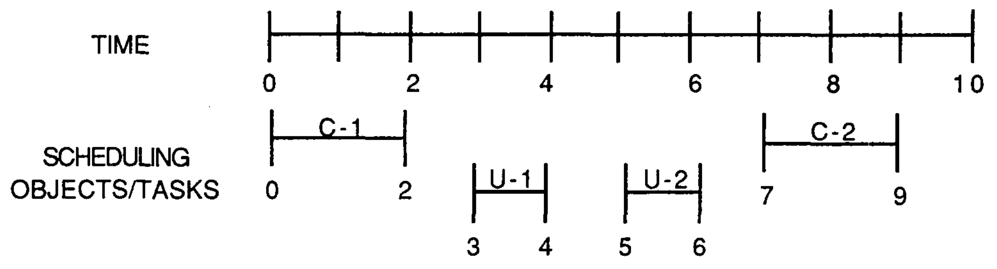

Figure 7. State Resources

Table 2. Table of State Resources

| Object | Abbreviation | Time | Possible States | Actual State | Desired State |
|---|---|---|---|---|---|
| Changer | C-1 | 0-2 | XX, YY, ZZ | YY | N/A |
| User | U-1 | 3-4 | AA, YY | N/A | AA |
| User | U-2 | 5-6 | AA | N/A | AA |
| Changer | C-2 | 7-9 | AA, ZZ | AA | N/A |

*Example 15. State Users*
Figure 7 ("State Resources") shows a timeline from time 0 to time 10, and four scheduling objects. Table 2 ("Table of State Resources") details the types of scheduling objects and the time, the state list, and the actual state (if a changer object) or desired state (if a user object) of each. C-1's duration is from 0 until 2, with the actual state being Mode YY. But another changer object does not appear until time 7, so the actual state from time 0 until time 7 is Mode YY. U-1 has a state list of AA and YY, but U-1 wants to use Mode AA and the actual state is set to Mode YY. U-1 can either change its state in use from Mode AA to Mode YY or move behind C-2. U-2 also desires Mode AA, but it has only one choice, and that's to move to another position in time where Mode AA is supported. Since C-2 sets Mode AA, U-2 may move to any time frame after the mode is changed to Mode AA.

Example 15 shows a changer object setting the actual state and illustrates that the actual state will remain the same, even after the conclusion of the changer object. The actual state will not change until another changer object is scheduled. This example also shows that conflicts will occur when changer and user states do not coincide within the same time frame. Note that the last three objects, U-1, U-2, and C-2, desire the same state.

## 4.5 SPECIAL RESOURCES

Special (or hybrid) resources are unique; therefore, they cannot be represented by simple combinations of previously mentioned resources. Digital tape recorders (DTRs) and batteries fall into this hybrid category. Below is a description of the special resources, along with an explanation of why these resources are unique.

### 4.5.1 The Digital Tape Recorder Resource

Digital tape recorders (DTRs) record and play back data gathered throughout the life of the mission. DTRs are similar to a depletable resource in that DTRs are resettable and dependent (upon the rate at which the data was recorded or played back). But, with DTRs, there are other considerations. Positioning of the tape is important: If the DTR is recording, care must be taken not to record over tape footage that has not been played back. DTR usage is constrained to follow the FIFO (first in, first out) rule when the DTR is recording and playing back. Scheduling usages for the DTR comprise an arena called *DTR management.*

### 4.5.2 The Battery Resource

Rechargeable batteries are a combination of several types of resources. These batteries store a resettable amount of energy that is available to the spacecraft. This aspect of batteries is represented as a depletable/resettable resource. The power output of the batteries contributes to the total power available to the spacecraft. This aspect of the battery is classified as a non-depletable resource.

*Example 16. The Rechargeable Battery*
Batteries usually output a steady amount of power until a minimum power threshold is reached. At this threshold, the output power drops drastically. For this example, the energy of the battery starts at 1kilowatt-hour and depletes at a rate of 50 watts (output power). The battery depletes at that rate until a minimum threshold of 150 watt-hours is reached. The output power decreases in proportion to the drop in the power threshold. In this model of the battery, energy is represented as a depletable resource, and the output power is represented as a non-depletable resource.

# SECTION 5

# DEPENDENCIES

This section discusses the term *dependency* as it refers to resources and sequencing elements.

## 5.1 RESOURCE DEPENDENCY

A *resource dependency* occurs when one or more resources directly influence another resource or resources.

*Example 17. E = P*T*
Energy utilization (energy = power • time) by a sequencing element is directly derived from the element's power usage and its duration. If the sequencing element ABC utilizes a constant 30 watts of power for its duration of 30 minutes, then the energy consumed by ABC is 15 watt-hours.

## 5.2 RELATIONSHIPS AMONG SEQUENCING ELEMENTS

Sequencing elements may be dependent upon one another. Elements may be chronologically dependent, and there may be delay- or separation-time requirements for two or more elements. There are many different types of element dependencies. The following sections describe the different sequencing-element dependency types [14].

### 5.2.1 Existence Dependency

An *existence dependency* is the most primitive and simplistic form of relationship among sequencing elements. An existence dependency occurs when an element can only exist in the sequence if another sequence element is already present in the sequence.

*Example 18. Existence Dependency*
Consider the following task as an example of an existence dependency: If Instrument X is in Mode 5, schedule a calibration activity. Existence dependency usually implies checking a certain condition of a resource. More stringent requirements build upon the primitive existence dependency and further refine the implicit temporal relationships between related sequencing elements.

### 5.2.2 Terminology for Describing Temporal Relationships

This section describes the terminology used when one is describing the temporal relationships of related elements [14]. The terms are *Comes-During, Not-During, Comes-Before, Starts-Before, Comes-After, Starts-With,* and *Ends-With.*

5.2.2.1 Comes-During. The expression "Comes-During" defines that a sequencing element may only be scheduled within the duration of a related sequencing element.

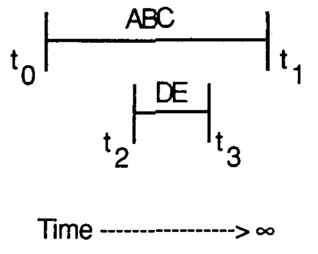

Figure 8. Comes-During

In Figure 8 ("Comes-During"), sequencing element DE Comes-During sequencing element ABC, which implies that $t_2$ to $t_3$ must be included within $t_0$ to $t_1$. Other terms for Comes-During are "concurrency," "non-synchronicity," and "gobbled" [14].

5.2.2.2 Not-During. Not-During is the opposite of Comes-During: A sequencing element may not be scheduled within the same duration as the related sequencing element. Similar terms for Not-During are "non-concurrency" and "avoidance" [14].

5.2.2.3 Comes-Before. The term "Comes-Before" is used to specify that one sequencing element must occur before another sequencing element in the sequence.

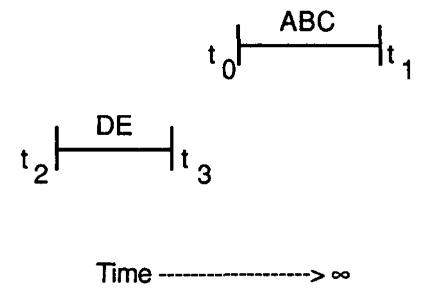

Figure 9. Comes-Before

An example of this is illustrated by Figure 9 ("Comes-Before"). Sequencing element DE Comes-Before sequencing element ABC; therefore, $t_3$ must be less than or equal to $t_0$. A similar term is "precedence" [14].

5.2.2.4 Starts-Before. The term "Starts-Before" refers to a sequencing element that starts before another sequencing element. In Figure 9, if DE Starts-Before ABC, then $t_0$ must be greater than $t_2$, but ABC can overlap DE.

5.2.2.5 Comes-After. The term "Comes-After" is used to specify that one sequencing element must occur after another sequencing element. This relationship is the inverse of the Comes-Before dependency.

In Figure 9 ("Comes-Before"), if ABC Comes-After DE, $t_0$ must be greater than or equal to $t_3$.

5.2.2.6 Starts-With. The term "Starts-With" defines that a sequencing element must start at the same time as the specified sequencing element. Figure 10 ("Starts-With") shows that ABC starts at the same time as DE; therefore, $t_0$ equals $t_2$.

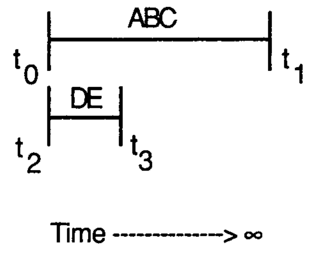

Figure 10. Starts-With

Note that the elements' stop times do not have to coincide.

5.2.2.7 Ends-With. The term "Ends-With" defines that a sequencing element must stop/end at the same time as the specified sequencing element. Figure 11 ("Ends-With") shows that ABC stops simultaneously with DE. This implies that $t_1$ equals $t_3$.

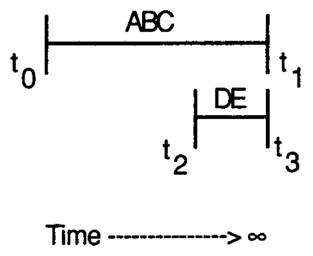

Figure 11. Ends-With

Note that the elements' start times do not have to coincide.

# SECTION 6

# SEQUENCING CONSTRAINTS AND CONSIDERATIONS

The process of determining where to schedule a sequencing element involves enforcing many constraints. Constraints and considerations that influence the placement of an element are

(1) Time specifications
(2) Sequence elements' dependencies, with timing requirements
(3) Scope of dependent sequencing elements
(4) Resource constraints
(5) Activity structures based upon resource profiles

All sequencing elements may not have to contend with each of the constraints mentioned above. This section will attempt to give the reader information about these constraints, which may be considered in spacecraft sequencing.

## 6.1 TIME SPECIFICATIONS

Some sequencing elements will have time specifications. Time specifications limit the flexibility for placing an element. After a sequence has been generated, all sequencing elements will have explicit start and stop specifications. The following two subsections describe start/stop times and window intervals.

### 6.1.1 Start and Stop Times

The most primitive and direct form of specification is assignment of specific start and stop times telling where to place the element into the sequence. When the start and stop times are specified, placement of the element into the sequence is not flexible. Usually start/stop times are not specified for elements until after the sequence is generated. But usually these elements have time window specifications.

### 6.1.2 Time Windows

A time window specification specifies temporal intervals (*windows*) where the element may be placed in the sequence. There may be a single time window where the placement of the element is legal or multiple legal windows. Windows still limit the flexibility of placement for the element, but the element does have more flexibility than if the explicit start/stop times were specified. For windows to be valid, a sequencing element's duration must be encompassed within the specified time windows.

## 6.2 ELEMENT DEPENDENCY CONSTRAINTS WITH TIMING REQUIREMENTS

A constraint that is difficult to enforce is a combination of sequence element dependencies and delay/separation requirements (see Sec. 5.2). These timing requirements come in two forms: The sequencing element may have defined a minimum and maximum delay requirement, or the elements may have a specification that says that the element must occur *x* number of times in the sequence. Both steps within an activity, as well as multiple occurrences of the activity or event, may have delay requirements.

### 6.2.1 Steps

The following example demonstrates an activity that is composed of steps and has both step dependencies and delay requirements.

*Example 19. Picture Activity*
Activity ABC takes a picture of Target A. This activity requires four steps:

| | |
|---|---|
| Step 1: | Set the camera to the proper shutter speed and |
| Step 2: | Point the camera to the correct position and wait from 10 seconds to 20 minutes, then |
| Step 3: | Gather data, then |
| Step 4: | Transmit data to ground within an hour after taking the picture |

Element Dependencies:

> Step 1 and Step 2 Comes-Before Step 3
> Step 3 Comes-Before Step 4

Delay Requirements:

> Between Step 1/Step 2 and Step 3,
>
> > Minimum delay is 00:00:10
> > Maximum delay is 00:20:00
>
> Between Step 3 and Step 4,
>
> > Minimum delay is 00:00:00
> > Maximum delay is 01:00:00

Activity ABC has two possible configurations with respect to the element dependencies. The sequences of steps are

| | |
|---|---|
| #1 | Step 1 --> Step 2 --> Step 3 --> Step 4 |
| #2 | Step 2 --> Step 1 --> Step 3 --> Step 4 |

With either configuration, delay requirements must be enforced between the steps. This activity gives an example of step dependency requirements that also have timing specifications.

### 6.2.2 Activities or Events

Not only are there delay times between steps of an activity, but there may also be delay times between multiple occurrences of an activity or multiple occurrences of an event. These delay times may be *specified* or *implied.*

*Example 20. Implied Delay Time*
The requirement that Activity B must occur once a day in the sequence is an example of an implied delay time. Example 21 (below) is an example of explicitly specified delay times.

*Example 21. Temporal-Separation Requirements*
Let's suppose that Activity A must occur five times in the sequence and that each occurrence must be separated by one hour. When the delay times are specified, as they are here, scheduling the activity becomes more difficult. Let's say that Activity A is scheduled to occur five times in a sequence and that each occurrence must be separated by one hour. But let's suppose further that in the five occurrences of Activity A, the third occurrence (Activity A3) has a resource conflict. This means that because of the conflict of Activity A3, the entire group (all five occurrences of

Activity A) must be moved in an attempt to alleviate the conflict of Activity A3. After moving the entire group by three hours, we now find that we have a conflict with Activity A4. Again the group must be moved in an attempt to alleviate the conflict. This iterative process will continue until all five occurrences of Activity A can be scheduled. An alternative is to modify the delay time between the elements slightly if the separation requirements are flexible. That way, the elements may be moved slightly, instead of being put to the drastic moves previously described.

## 6.3 THE SCOPE OF DEPENDENT SEQUENCING ELEMENTS

When sequencing elements have other elements dependent upon them, this dependency may assist in pruning down the search space, or the dependency may cause ripple effects throughout the sequence. The two cases are described below.

### 6.3.1 Derived Windows

When a sequencing element is dependent upon another element, new windows based upon the primary element may be derived for the dependent element. Deriving these new windows for the dependent element will assist in restricting the time windows where it is legal to place the dependent element.

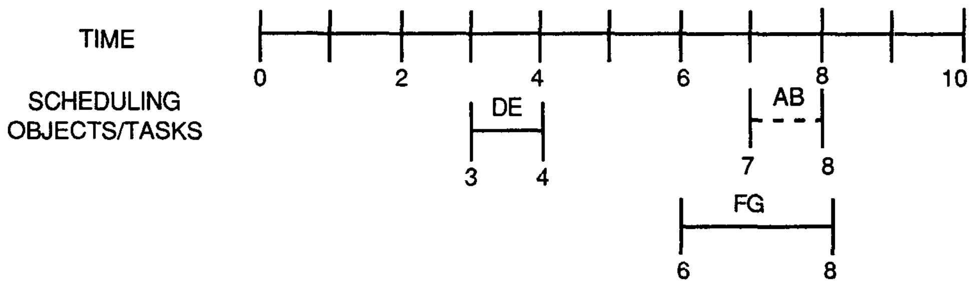

Figure 12. Derived Windows

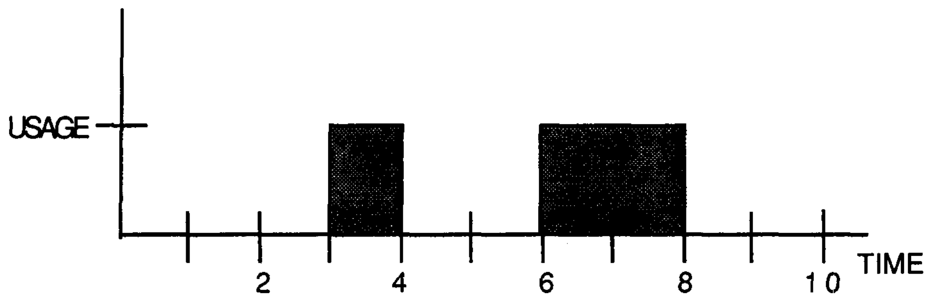

Figure 13. Resource Usage for the Derived Windows in Figure 12

*Example 22. Derived Windows*
Let's say that Figure 12 ("Derived Windows") shows two elements scheduled, with a third element waiting to be scheduled. The scheduled elements are element DE, scheduled for time units 3 to 4, and FG, scheduled for time units 7 to 8. Element AB has time windows from units 0 to 10 and waits to be scheduled for units 7 to 8. Element AB Comes-After element DE. Scheduling AB at units 7 to 8 would create a conflict because, looking at Figure 13 ("Resource Usage for the Derived Windows in Figure 12"), we can see that the resource is already allocated for time units 7 to 8. Note that AB has two constraints: a resource dependency and an element dependency. The areas for which resources are not allocated are time units 0 to 3, 4 to 6, and 8 to 10. Since AB

has a window of time units 0 to 10, AB can move to any one of the three available times. But, since AB is dependent upon DE, AB should actually be placed in an available time window after DE's stop time, which is between time units 4 to 10. To derive new windows for AB from the resource constraint and the dependency constraint, we intersect time windows 0 to 3, 4 to 6, and 8 to 10 with 4 to 10. The new derived windows that AB can use are times 4 to 6 or 8 to 10. Assuming that AB moves to time unit 4 to 5, let's check the element dependency relationship, DE Comes-Before AB, and the resource utilization. It appears that AB can be scheduled without conflict in time interval 4 to 5. This is an example of how windows may be derived, based upon the resource- and element-dependency relationships, to scope the search space.

### 6.3.2 Ripple Effects

When a sequencing element is dependent upon another element, *ripple effects,* or cascading actions, may occur if the dependency relationship is not enforced during the sequencing action. Let's use the example in Section 6.3.1 (above) with identical initial conditions: The areas to which resources are not allocated are time units 0 to 3, 4 to 6, and 8 to 10. Initially, if the element dependency relationship, DE Comes-Before AB, is ignored, AB can be placed at the earliest time unit: 0 to 1. Resource constraints are not then violated, but the dependency relationship is violated. To enforce the dependency relationship, we try moving DE before AB. This cannot be done because the sequence starts at time 0. Next, we try moving AB to time interval 1 to 2. Again, the resource constraint is not violated, but the dependency constraint is, so we try moving DE to time interval 0 to 1. Moving DE is acceptable because both the resource and dependency constraints are enforced. Note that when the dependency constraint was not enforced initially, a ripple effect generated additional scheduling actions affecting other dependent elements.

## 6.4 RESOURCE CONSTRAINTS

As described in Section 4 ("Resources"), sequencing elements must use or be supported by resources in order to perform their tasks. The allocation of these resources also contributes to the placement of a task. When resources are allocated to a task, the resources must not be oversubscribed. If the resources are oversubscribed, a resource conflict exists. Resource constraints have a major influence in determining the start and stop times of sequencing elements.

*Example 23. Placement Derived From Resource Constraints*
Activity ABC uses 30 watts of power and has a duration of four time units. ABC has a time window from time interval 0 to 12. Where can ABC be scheduled so as to meet the temporal resource constraints defined in Figure 14 ("Power Allocation")? The Figure 14 graph depicts the maximum power allocation (bold line) and the power usage (light gray areas) with respect to time. Table 3 ("Description of Power Allocation") is the temporal breakdown of the allocations, usages, and the amounts not being allocated. From the table, it's apparent that Activity ABC, which uses 30 watts, can fit into two time frames: from times 1 through 4 and 6 through 12. But the duration of ABC is four time units, so the only time frame that ABC can fit is time 6 through 12.

This is an example of how resource allocations can influence the placement of a sequencing element.

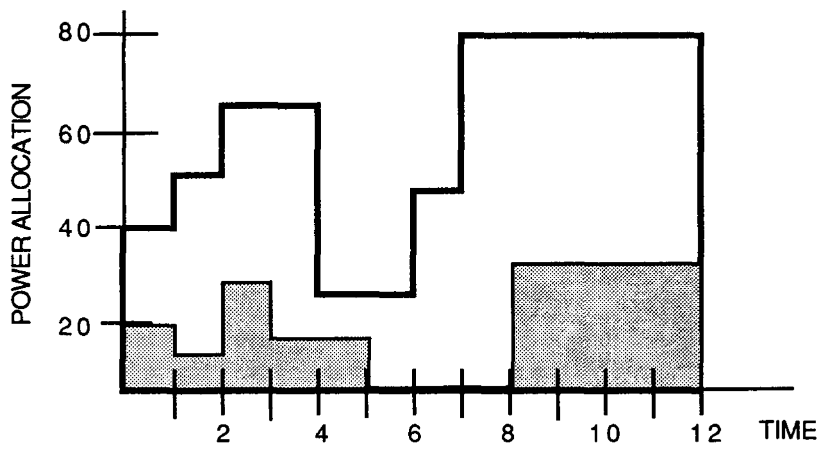

Figure 14. Power Allocation

Table 3. Description of Power Allocation

| Time | Maximum Allocation | Usage | Not Allocated |
|---|---|---|---|
| 0-1 | 40 | 20 | 20 |
| 1-2 | 55 | 10 | 45 |
| 2-3 | 65 | 30 | 35 |
| 3-4 | 65 | 15 | 50 |
| 4-5 | 25 | 15 | 10 |
| 5-6 | 25 | 0 | 25 |
| 6-7 | 50 | 0 | 50 |
| 7-8 | 80 | 0 | 80 |
| 8-12 | 80 | 34 | 46 |

## 6.5 ACTIVITY STRUCTURE WITH RESPECT TO RESOURCES

Activity structures may be effected by resource constraints/utilization. When the composition of an activity is flexible (has multiple configurations), the choice of configuration may be dependent upon the temporal resource-utilization profiles. In addition, multiple-occurrence activities may be defined by a looping cyclic (see Sec. 3.4.2.2), which saves on resource utilization.

### 6.5.1 Replanning

The actual step makeup of activities in the sequence is either predefined and therefore inflexible, or dependent on the temporal resource-utilization profiles. If the resource profile won't allow for a particular sequencing element's configuration, the sequencing element configuration must change to one that doesn't cause conflicts with the resources the element utilizes.

*Example 24. Replanning*
Activity XYZ, an oven experiment, cooks an object to test its own thermal features in zero gravity. To perform the experiment, we prefer to use Oven A at 500 degrees for three time units or Oven B at 400 degrees for four time units. The time window is between time units 2 and 11.

Activity XYZ prefers to use Oven A. Checking the resource allocation for Oven A, which is given in Figure 15, we find that Oven A is in use from time 1 until time 9. For Oven A to be used, the experiment must take place after time 9, when the oven is not in use. But if we are to use Oven A, we must use it for three time units and use it during the window of time interval 2 through 11. So it's apparent that Oven A cannot be used for this experiment. An alternative solution is to use Oven B. Looking at Figure 16 ("Oven B Usage"), we see that Oven B is not in use from time 5 to time 10. So, based upon the resource constraints, instead of scheduling XYZ with Oven A, we use Oven B, which uses four time units and may be used between time units 5 and 10. This is an example of how the configuration or planning of an activity is changed based upon resource constraints.

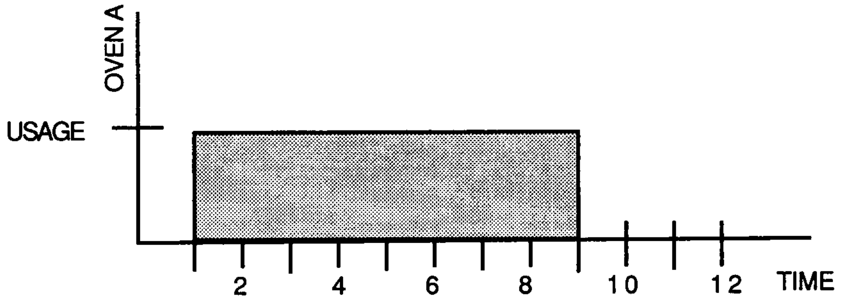

Figure 15. Oven A Usage

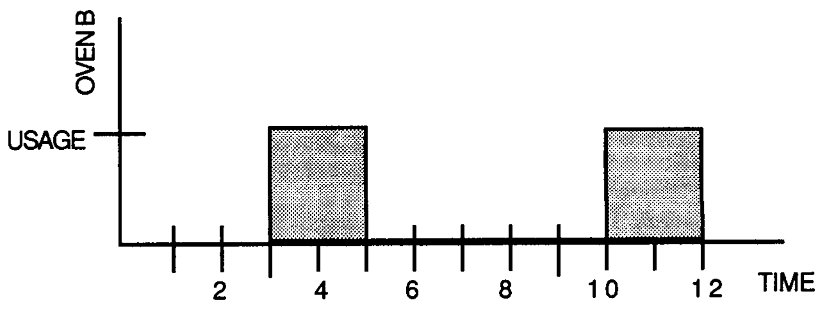

Figure 16. Oven B Usage

## 6.5.2 Looping Cyclics

Multiple-occurrence activities may be defined by looping cyclics (see Sec. 3.4.2.2), which saves on resource utilization. A looping cyclic is a predefined temporal-model definition of a sequencing element; it maximizes the utilization of resources.

*Example 25. A Looping Cyclic*
Activity ABC requires 20 memory bytes each time it occurs. Since ABC occurs five times in the sequence, the memory utilization is 100 memory bytes. However, if ABC is defined as a looping cyclic, the base cost is 20 memory bytes plus 16 additional memory bytes for the timing-and-repetition definition, for a total of 36 memory bytes. (If the same looping cyclic is called again without any changes, it will cost two memory bytes to invoke the cyclic.) The total saving of memory bytes is the original 100 minus 36, which is a saving of 64 memory bytes.

This example demonstrates how looping cyclics save on resource utilization.

# SECTION 7

# SEQUENCE-QUALITY DETERMINATION

Statistics and measurements are used during and after the sequencing process to assess the relative merits of the sequence. Sometimes these statistics and measurements are referred to as defining the "goodness" of a sequence. Many questions arise when one is determining the goodness of a sequence. Are all of the requests scheduled? Are all of the conflicts resolved? If compromises were made, are all the requesters satisfied? Are the resources being utilized in an efficient manner? This section addresses these and other questions and describes different sequence-measurement methodologies.

## 7.1 AGREEMENT ON THE SEQUENCE (MEETING OBJECTIVES)

One of the most important ratings of sequence quality is whether all parties involved with the sequence agree that it accomplishes their objectives. These parties include requesters and sequence-implementation personnel [6]. Usually more activities are requested than can be scheduled, owing to resource availability. During the sequence-integration process, many compromises are made by different requesters to help alleviate conflicts. Some requesters may shorten the duration of their activity to get more experiment time later; others may decide that gathering data on Target X is more important than gathering data on Target Y, so Target Y is eliminated; others may compromise by merging their experiment requests. Many compromises are made to help generate a conflict-free sequence. After a sequence is generated, all parties involved must agree that the sequence accomplishes their objectives. If all parties are satisfied, then there is agreement on the sequence [12].

## 7.2 SEQUENCE SAFETY

Another measurement is sequence integrity and safety. After a sequence is generated, conflicts over some constraints may still exist, and those conflicts must be assessed. If the safety margin allows the conflict to exist without harm to the spacecraft and the sequence implementers agree that the conflict can exist, then the constraint may be waived.

## 7.3 PERCENTAGE OF SUCCESSFUL REQUESTS

Usually there are more requests than can be accomplished or scheduled in a sequence. Another rating of a sequence's goodness is the percentage of requests that is scheduled into the final sequence versus the number of requests originally sought. This rating may be biased because some of the unscheduled requests may be scheduled into future sequences.

## 7.4 RESOURCE UTILIZATION

Resource utilization indirectly influences determination of sequence goodness. Statistics on the distribution of resources are used to influence future sequences. This is particularly true for resources that do not impact a specific sequence, but are concerned with the overall mission or project. It is usually important to stretch out the use of these resources (minimize the waste) over time to ensure that the spacecraft's useful lifetime is as long as is necessary to ensure mission completion. For example, a digital tape recorder (DTR) is designed to perform up to 50,000 start/stop cycles with a confidence level of 99 percent. The actual number of start/stop cycles for the DTR may not be important in the sequences for the early portions of the mission, but DTR overutilization early on may influence the later stages of the mission. This, in turn, may influence the goodness of later sequences.

## 7.5 SEQUENCE FLEXIBILITY

Some spacecraft sequencing domains permit real-time dynamic changes to the sequence through real-time modifications to the sequence. This implies that the sequence must be flexible. Let's use data gathering as an example. To gather data, a hypothetical instrument may use one of two modes: Mode A for three seconds or Mode B for eight seconds. The decision of which mode to use is made during sequence execution and before the activity is to be executed. For the decision to be made, the sequence must be flexible enough to ensure that during a time frame of three to eight seconds, either mode may be utilized. This ability to choose modes is an example of real-time commanding.

Another example is a last-minute trajectory update. If an instrument must be slewed, the duration of the slew will be dependent upon the trajectory updates. The sequence must be flexible to allow for an increased duration of the slew. Allowance for these types of dynamic real-time changes must be pre-scheduled into sequences to ensure their goodness and flexibility.

## 7.6 SEQUENCE TRANSITION

Since many missions have contiguous sequences, it is important to end a sequence according to another criterion of goodness — that the transition to the following sequence be smooth. If a sequence is left in a "bad" state, additional activities may be required to ready the resources for the following sequence. This may result in reducing the percentage of successful requests to the following sequence. For example, if a sequence has concluded without all the DTR data being played back, the following sequence will have to play back the data before additional data can be recorded. This results in the following sequence utilizing time to play back data — time that could be otherwise used to execute outstanding requests.

## SECTION 8

## GLOSSARY*

*Activity.* An observation or task that a science or engineering representative would like to have performed on board the spacecraft. An activity consists of temporal definitions as well as resource usages.

*Activity* (Planner). A Planner activity is related to an activity type in that the activity is created by giving values to the arguments of an activity type [14].

*Activity Type* (Planner). A Planner activity type is a generic sequence component without specified parameter values [14].

*Apart* (Planner). This dependency specifies that the absolute value of the difference between A's start time and B's start time is at least *x* time units, where A and B are both steps or both activities [14]: (See Sec. 6.2, "Element Dependency Constraints With Timing Requirements.")

*Availability Resource* (Spacelab). The resources in this category represent the existence or non-existence of an item or action. The current state of the resource is determined by the spacecraft and/or instrument orientation and geometric relationships, or the current state is based upon generating or not generating some type of environmental effect that may influence other activities [17]. Synonym for *synchronicity resource.*

*Avoidance* (Planner). A's time span does not intersect B's, where A and B are both steps or both activities [14]. Synonym for *Not-During* (q.v.).

*Block* (Voyager, Galileo). A group consisting of commands, events, or each of these and having defined time interrelationships among them, a block performs a system-level function or activity. A block may describe either a complete activity or an activity element that is often used in creating spacecraft sequences. The application of a block is defined by options and/or parameters [15].

*Case-Activity.* An activity that has multiple configurations of steps. All of these configurations are predefined.

*Combined Resources.* A group of resources that is complex because of the dependencies between the resources. (See *resource dependency.*)

*Comes-After.* One sequencing element must occur after another sequencing element.

*Comes-Before.* One sequencing element must occur before another sequencing element. (See *precedence.*)

*Comes-During.* A sequencing element may only be scheduled within the duration of a related sequencing element. (See *concurrency* and *gobbled.*)

* *Some of the definitions in this section refer to existing sequencing tools and specific projects. See the listed references for more information on these applications.*

*Command* (Galileo). On Galileo, a command is issued by the CDS to either the CDS or AACS or to other spacecraft systems or instruments as part of a sequence [16].

*Command* (Voyager). Same definition as *event* (q.v.) [7].

*Concurrency* (Spacelab). Scheduling one step of an experiment to occur at the same time as a step of another experiment [17].

*Constraint.* A relationship or limit that must be checked to validate the sequence. Constraints are in the form of resources, mission rules, and flight rules.

*Consumable Resource.* Synonym for *depletable resource* (q.v.).

*Contextual Control.* Users gain contextual control by grouping the sequence elements together to perform editing actions.

*Cumulative Resource* (Planner). A cumulative resource is one that is not measured pointwise. To compute the "net usage" of a cumulative resource up to a given point in time requires knowing about the activities that are active at and before the time, the states in effect at and before the time, and the initial conditions or resettings of the resource [14]. Synonym is *depletable resource* (q.v.).

*Cyclic* (Voyager). A temporal link or activity that takes the form of a subroutine. A cyclic is created to represent an activity that is called many times. The creation of a cyclic saves on resource usage (specifically, CCS words) [7].

*Deep Space Network* (DSN). NASA's worldwide ground-based radio communications network that (1) provides two-way communications for spacecraft exploring the solar system and (2) performs science experiments on extraterrestrial radio sources.

*Dependencies.* See *resource dependency* and *sequence element dependency.*

*Depletable Resource* (Spacelab). The depletion rate of these resources is dependent on both time and the rate of consumption. Some depletable resources are resettable and/or replenishable resources. Depletable resources keep a running total of the resource allotment. The resource allotment decreases with usage over time. Synonym is *cumulative resource* (q.v.).

*Ends-With.* A sequencing element must stop/end at the same time as the specified sequencing element. The elements' start times do not have to coincide.

*Event.* A task or experiment containing information such as start/stop times and resource usages. An event is a stand-alone structure with no other events dependent upon it. A majority of events are used for information purposes. Synonym is *command* (q.v.).

*Experiment* (Spacelab). A collection of models that, when executed, adds to the body of science information [18].

*Experiment Scheduling* (Spacelab). Experiment scheduling for Spacelab is primarily a matter of positioning the experiments in the mission in such a way that they do not interfere with each other and can collect the desired data [18].

*Flight Rules.* Spacecraft and instrument rules and constraints that pertain to the spacecraft's health and safety.

*Gobbled* (Planner). A dependency specifying that A's time span is contained in B's, where A and B are both steps or both activities [14]. Synonym for *Comes-During* (q.v.).

*Group* (Planner). A list of steps and/or activities, used during a session for ease of manipulating several steps and/or activities at once [14].

*Group of Operational Envelopes* (Telescience). A combination of operational envelopes that will accomplish the desired observation. (See *operational envelope.*)

*Initial/Final State.* The initial and final state of all of the resources applicable to the sequence.

*Link* (Communication). "Link" refers to a communication link between the spacecraft and ground, i.e., the uplink and downlink [7]. This is also the standard definition of "link."

*Link* (Voyager, Galileo). A descriptively named segment of time to which spacecraft and ground resources are allocated to perform a spacecraft activity. A link is used in the uplink-design process as the most summary level of sequence-component description to allow people to understand the request form [15].

*Load* (Galileo). The sequence of binary data as received by the spacecraft [16].

*Looper* (Galileo). A series of commands and associated parameters that performs a sequencing function. A characteristic of a looper is that the commands and parameters are repeated several times. In this way, a sequence event may be programmed to occur several times at a CDS memory word cost approximately equal to the cost of only one execution [16].

*Looping Cyclic* (Voyager). A cyclic with temporal constraints. Similar to *looper* (q.v.).

*Meta-Activity.* An abstract collection of activities that is based on a structural implementation methodology. There are four types of meta-activities: *cyclics, looping cyclics, blocks,* and *groups* (q.v.).

*Mission Sequencing Table* (Viking). An activity (q.v.) on Viking [7].

*Model* (Spacelab). The database representation of an experiment or part of an experiment. A model is a collection of constraint and execution definitions. Some of the definitions apply to the whole model, and some apply to the steps of the model [18].

*Near* (Planner). A dependency in which the absolute value of the difference between A's start time and B's start time is at most $x$ time units, where A and B are both steps or both activities and $x$ is a positive integer. If A and B are activity types, this is to be interpreted as meaning that for every instance of A there is an instance of B whose start time is at most $x$ time units from A's start time, and vice versa [14]. (See Sec. 6.2, "Element Dependency Constraints With Timing Requirements.")

*Non-Consumable Resource.* Synonym for *non-depletable resource* (q.v.).

*Non-Depletable Resource.* A resource that may be utilized up to its maximum allocation for a time frame. The resource usage is cumulative by multiple users unique to each time frame pertaining to that resource. After a task has concluded, the non-depletable resource that the task was utilizing will again be available for use by other tasks. Synonyms are *pointwise resource* (q.v.) and *pooled resource.*

*Not-During.* A sequencing element may not be scheduled within the same time frame as the related sequencing element. Synonym for *avoidance* (q.v.).

*Observation.* Synonymous with *activity* (q.v.).

*Operational Envelope* (Telescience). An OE combines resource information, temporal relations, priority assignments, and ownership information into one entity. The use of envelopes allows grouping of resources so that the scheduling system may work at a higher level — that of allowing greater atomicity in scheduling requests [19]. (See *group of operational envelopes.*)

*Pointwise Resource* (Planner). A pointwise resource is one whose total usage at a given time depends only on the steps whose time spans contain that time [14]. Synonym for *non-depletable resource* (q.v.).

*Pooled Resource.* Synonym for *non-depletable resource* (q.v.).

*Precedence* (Planner). A dependency defining that the end time of A is no later than the start time of B, where A and B are both steps or both activities [14]. Synonym for *Comes-Before* (q.v.).

*Principal Investigator.* A representative from a college or university who represents a group of scientists who are involved in instrument or experiment development. This representative ensures that the objectives of the experiment are met during and after the sequence-generation process.

*Profile Activity* (Galileo). A descriptively named science, navigation, or engineering activity that usually involves several spacecraft and/or ground (sub)systems [15].

*Project Representatives* (Galileo). To help centralize the interface between the Project staff and the technical divisions, the technical divisions provide a project representative, who coordinates the division's total technical and administrative activities related to the Project [15].

*Request.* A science or engineering proposal to perform an activity or experiment on board the spacecraft. The request consists of a description of the proposal, the request's temporal location, and the resources to be utilized.

*Requesters.* Engineers and scientists who are petitioning for their experiment/activity to be performed on board the spacecraft. Engineers are concerned with the health and safety of the spacecraft, whereas scientists are concerned with gathering data for their science experiments.

*Resource.* Any available commodity or means that may be allocated to the accomplishment of a task [15].

*Resource Dependency.* One or more resources directly influence one or more other resources. (See *combined resources.*)

*Resource Envelope* (Telescience). Consists of a list of resources and the amount of each resource allocated to a payload that has been given that envelope [19].

*Resource Profile.* The temporal variation of a resource's usage and maximum allocation.

*Reusable Resource.* Synonym for *non-depletable resource* (q.v.).

*Schedule.* A procedural plan that indicates the timing and sequence of tasks to perform on board the spacecraft.

*Sequence* (Planner). A collection of activities and their constituent steps [14].

*Sequence* (Voyager, Galileo). A stored series of commands and calls to the spacecraft's expanded blocks, or to any of the mnemonic forms generated in the process of developing that sequence. A sequence is a load of binary data in the sequence memory of the spacecraft.

*Sequence Boundaries.* The predetermined start and stop times of a sequence.

*Sequence Component* (Galileo). A logical unit of spacecraft activity that is placed within a sequence and performs a defined task. It is used as a sequence-development tool to define the sequence in progressively greater detail [16].

*Sequence-Element Dependency.* Sequencing elements may be dependent upon one another. Elements may be chronologically dependent, and there may be delay or separation timing requirements between two or more elements.

*Sequencing* (Voyager). The process of defining and scheduling activities.

*Slew.* Changing the positioning of an instrument/platform.

*Starts-Before.* A sequencing element starts before another sequencing element, and the elements can overlap.

*Starts-With.* A sequencing element must start at the same time as the specified sequencing element. The elements' stop times do not have to coincide.

*State Resource.* This category represents resources that have modes or conditions of being. Some tasks set the mode of a state, while other tasks check to see if they can use the mode that is currently set. Other tasks prohibit certain states from occurring within selected time frames.

*Steps* (PLAN-IT). Here, steps represent actions that occur in a sequence. Steps may have temporal resource dependencies and temporal step dependencies. A group of steps that are dependent upon each other is an *activity* (q.v.).

*Steps* (Planner). In Planner, an activity is composed of steps at a lower level than the specified activity; ultimately, each activity is composed of steps. A step is the atomic planning unit in Planner [14].

*Steps* (Spacelab). In Spacelab, steps make up a clearly delineated portion of the procedures associated with an experiment: They are a part of a model. Steps are generally executed sequentially, but not necessarily contiguously. Each step may have resource requirements, delays relative to other steps of the model, and target and attitude requirements [18].

*Synchronicity Resource.* Synonym for *availability resource* (q.v.).

*Target Windows.* The time windows when the target is in view of the spacecraft. Synonym for *view windows.*

*Time Frame.* Synonym for *time span* (q.v.).

*Time Span* (Planner). The time span of a step is the interval of time between the start time and the end time of the step. The time span of a sequence is the interval of time between the start time of the step that starts earliest and the end time of the step that ends latest. The time span of an activity is the union of the time spans of all the steps that constitute the activity [14]. Synonym for *time frame.*

*Time Windows* (PLAN-IT). A list of time intervals (start and stop times) that defines the valid temporal locations of a sequence component. Synonym for *windows.*

*Time Windows* (Planner). A list of pairs of absolute times, each pair denoting an interval of time [14]. Synonym for *windows.*

*Timed Precedence* (Planner). The end time of A is at least (or at most, or equal to) *x* time units earlier than the start time of B, where A and B are both steps or both activities and *x* is a positive integer [14]. (See Sec. 6.2, "Element Dependency Constraints With Timing Requirements.")

*Timeline* (Voyager, Galileo). A graph with time on the abscissa, showing the scheduled timing of activities [16].

*View Windows.* Time windows when the target is in view of the spacecraft. Synonym for *target windows.*

*Waiver.* An agreed-upon acceptance of a particular constraint violation in the sequence.

*Windows.* See *time windows* (PLAN-IT) and *time windows* (Planner).

## SECTION 9

## REFERENCES

[1] Katz, B., Brooks, R.N., "Understanding Natural Language for Spacecraft Sequencing," *Journal of the British Interplanetary Society*, Vol. 40, 1987.

[2] Finnerty, D.F., Martin, J., Doms, P.E., "ASSET: An Application in Mission Automation for Science Planning," *Journal of the British Interplanetary Society,* Vol. 40, 1987.

[3] Dias, W.C., Henricks, J.A., Wong, J.C., "PLAN-IT: Scheduling Assistant for Solar System Exploration," *Telematics and Informatics*, Vol. 4, Number 4, 1987.

[4] Berner, C.A., Personal Interview, Jet Propulsion Laboratory, February 21, 1989.

[5] Eggemeyer, W.C., Bowling, A., "Deep Space Network Resource Scheduling Approach and Application," *Proceedings of Space Applications of AI and Robotics Conference*, Goddard Space Flight Center, Greenbelt, Md., May 1987.

[6] Brooks, R.N., Personal Interviews, Jet Propulsion Laboratory, January 12, 1989, and February 16, 1989.

[7] McLaughlin, B., Personal Interview, Jet Propulsion Laboratory, January 18, 1989.

[8] Grenander, S.U., "Applications of Artificial Intelligence to Unmanned Spacecraft Missions," American Institute of Aeronautics and Astronautics (AIAA) Conference, Fall 1986.

[9] Eggemeyer, W.C., Grenander, S.U., "PLAN-IT Applications and Knowledge Gained," *Proceedings of Workshop on Operations Planning and Scheduling Systems for the Space Station Era*, University of Colorado–Boulder, Boulder, Colo., August 1987.

[10] Wolff, D.M., "CRAF Dummy Sequence Requests for PLAN-IT," Interoffice Memorandum No. 315.2-86-005 (internal document), Jet Propulsion Laboratory, July 25, 1986.

[11] Bahrami, K., et al., "Space Power System Scheduling Using an Expert System," *Proceedings of 21st Intersociety Energy Conversion Engineering Conference*, San Diego, Calif., August 1986.

[12] Biefeld, E., "PLAN-IT: Knowledge-Based Mission Sequencing," *Proceedings of Advances in Intelligent Robotics Systems Conference*, Cambridge, Mass., October 1986.

[13] Brooks, R.N., "The Evolution of the Voyager Mission Sequence Software and Trends for Future Mission Sequence Software Systems," AIAA 26th Aerospace Sciences Meeting, January 11-14, 1988, Reno, Nev.

[14] Starbird, T., *Space Flight Operations Center Sequence Subsystem (SEQ) Planning Program Set Software Requirements Document,* Jet Propulsion Laboratory, SSEQ0006-01-00-07 (internal document), May 6, 1988.

[15] Bouvier, H.K., *Project Galileo, Software, Technical and Administrative Procedures, Glossary of Terms and Acronyms*, Jet Propulsion Laboratory, 625-510, Vol. II, STAP 1.1 (internal document), April 30, 1980.

[16] Zawacki, S.J., *Galileo Mission Operations System, Functional Requirements, Mission Sequence System*, Jet Propulsion Laboratory, 625-500, Vol. I, MOS-GLL-4-211B (internal document), September 9, 1982.

[17] "Experiment Scheduling System, Session 6," Tutorial Session for Users of ESP, NASA/Marshall Space Flight Center, Huntsville, Ala.

[18] Japp, J., "Mission Timeline Analysis Demonstration," NASA/Marshall Space Flight Center, June 3, 1986.

[19] Lewis, E., *Telescience Demonstration, Scheduling System Concepts and Requirements and Prototype Scheduling System Concepts and Requirements,* Version 1.1, February 27, 1987.

TECHNICAL REPORT STANDARD TITLE PAGE

| | | |
|---|---|---|
| 1. Report No. 89-25 | 2. Government Accession No. | 3. Recipient's Catalog No. |
| 4. Title and Subtitle<br>A Planning and Scheduling Lexicon | | 5. Report Date<br>September 15, 1989 |
| | | 6. Performing Organization Code |
| 7. Author(s)<br>Jennifer W. Cruz and William C. Eggemeyer | | 8. Performing Organization Report No. |
| 9. Performing Organization Name and Address<br>JET PROPULSION LABORATORY<br>California Institute of Technology<br>4800 Oak Grove Drive<br>Pasadena, California 91109 | | 10. Work Unit No. |
| | | 11. Contract or Grant No.<br>NAS7-918 |
| | | 13. Type of Report and Period Covered<br>JPL Publication |
| 12. Sponsoring Agency Name and Address<br>NATIONAL AERONAUTICS AND SPACE ADMINISTRATION<br>Washington, D.C. 20546 | | 14. Sponsoring Agency Code<br>RE 4 BP-450-17-02-01-00 |
| 15. Supplementary Notes | | |

16. Abstract

This publication focuses on mission planning and scheduling for spacecraft. Planning and scheduling work is known as sequencing. Sequencing is a multistage process of merging requests from both the science and engineering arenas to accomplish the objectives defined in the requests. The multistage process begins with the creation of science and engineering goals, continues through their integration into the sequence, and eventually concludes with command execution on board the spacecraft.

The objective of this publication is to introduce some formalism into the field of spacecraft sequencing-system technology. This formalism will make it possible for researchers and potential customers to communicate about system requirements and capabilities in a common language. This publication is not intended to be the definitive work describing all capabilities and requirements; it is meant to lay a good enough foundation to allow follow-on development.

| | |
|---|---|
| 17. Key Words (Selected by Author(s))<br>121. Ground Support Systems and Facilities (Space)<br>124. Spacecraft Communications, Command and Tracking<br>239. Operations Research | 18. Distribution Statement<br>Unclassified; unlimited |

| 19. Security Classif. (of this report) | 20. Security Classif. (of this page) | 21. No. of Pages | 22. Price |
|---|---|---|---|
| Unclassified | Unclassified | 43 | |

JPL 0184 R 9/83